AMENDMENTS TO THE CLAIMS:

This listing of the claims will replace all prior versions, and listings, of the claims in this application.

Listing of Claims:

1. (Previously Presented) A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:

extracting a first codeword of H bits from the bitstream;

modifying the codeword by shifting it by a first shift value;

using this modified codeword to identify using at least a first data structure either a symbol or a second different data structure having a second offset value and a second shift value; and

if a second data structure is identified using the first data structure:

modifying the codeword by subtracting the second offset value and shifting the result by the second shift value; and

using this modified codeword to identify using the second data structure either a symbol or a third different data structure having a third offset value and a third shift value.

2. (Previously Presented) A method as claimed in claim 1, further comprising accessing a look-up table to obtain the first shift value and accessing the look-up table to obtain the second offset value and the second shift value.

3. (Previously Presented) A method as claimed in claim 1, wherein the first data structure represents a first level of the Huffman coding tree and the second data structure represents a second, lower level of the Huffman coding tree.

4. (Previously Presented) A method as claimed in claim 1, further comprising receiving at least a value of H, the first shift value, the second offset value, the second shift value, the first data structure and the second data structure.

5. (Previously Presented) A method as claimed in claim 1, wherein the step of modifying the codeword by shifting it by the first shift value comprises firstly subtracting a first off-set value, if any, from the codeword and then shifting the result by the first shift value.

6. (Cancelled)

7. (Previously Presented) A storage medium or transmission medium embodying a computer program for performing the method of claim 1.

8. (Previously Presented) A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:

extracting a codeword of H bits from the bitstream;

modifying the codeword by firstly subtracting a first off-set value, if any, from the codeword to obtain a result and then shifting the result by a predetermined shift value; and

using the modified codeword to identify a symbol using at least a first data structure.

9. (Previously Presented) A method as claimed in claim 8, further comprising accessing a look-up table to obtain the predetermined shift value.

10. (Previously Presented) A method as claimed in claim 8, wherein the first data structure represents a first level of the Huffman coding tree.

11. (Previously Presented) A method as claimed in claim 8, further comprising receiving at least the value of height H, the predetermined shift value, and the first data structure.

12. (Cancelled)

13. (Cancelled)

14. (Previously Presented) A storage medium or transmission medium embodying a computer program for performing the method of claim 8.

15. (Previously Presented) A decoder for decoding a bitstream encoded according to a Huffman coding tree of height H comprising:

a memory for storing a plurality of data structures representing the Huffman coding tree of height H including at least a first data structure having an associated first offset value and an associated first shift value and a second data structure having an associated second offset value and an associated second shift value; and

a processor operable to subtract an offset value from a codeword of H bits taken from the bitstream;

shift the result by a shift value; and

address a data structure using the shifted result.

16. (Original) A decoder as claimed in claim 15, wherein the first data structure represents a first level of the Huffman coding tree and the second data structure represents a second, lower level of the Huffman coding tree.

17. (Original) A decoder as claimed in claim 16, wherein the first shift value corresponds to the first level.

18. (Previously Presented) A decoder as claimed in claim 16, wherein the second shift value corresponds to the second level.

19. (Previously Presented) A decoder as claimed in claim 16 wherein the second offset value identifies a position of a first sub-tree within the Huffman tree.

20. (Previously Presented) A decoder as claimed in claim 17, wherein the processor is operable having obtained a value from addressing the associated data structure, to perform a comparison using that value and in dependence upon the comparison either use the value to

identify a symbol or a new current offset value.

21. (Original) A decoder as claimed in claim 20, wherein the comparison uses the MSB of the value.

22. (Previously Presented) A decoder as claimed in claim 20, wherein the current offset value is initially set to the first offset value.

23. (Previously Presented) A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:
storing a first data structure comprising a value for each possible node at a first level of the tree;
storing a second data structure comprising a value for each possible node within a first sub-tree at a second, higher level of the tree;
extracting a first codeword of H bits from the bitstream;
converting a value of the first codeword into a first node position within the tree at the first level of the tree; and
accessing the first data structure to obtain the value corresponding to the first node position, wherein that value refers to the second data structure;
converting the value of the first codeword into a second node position within the first sub-tree at the second level of the tree; and
accessing the second data structure to obtain the value corresponding to the second node position.

24. (Cancelled)

25. (Previously Presented) A storage medium or transmission medium embodying a computer program for performing the method of claim 23.

26. (Previously Presented) A method of decoding a codeword from a bit stream comprising:

receiving a representation of a Huffman tree as a plurality of ordered data structures comprising: a first data structure associated with an identified first level L1 of the tree and comprising a plurality of data entries, each entry corresponding to a node of a full tree at the identified first level and at least a second data structure associated with an identified second level L2 of the tree and with an identified first sub-tree and comprising a plurality of data entries, each entry corresponding to a node of the first sub tree, when full, at the second identified level;

obtaining a value for a first level L1 in a Huffman tree identifying the node in the first level L1 of the tree, when full, corresponding to the first L1 bits of the codeword;

obtaining from the first data structure a data entry for the identified node, that identifies a further data structure or identifies a symbol; and

if the data entry identifies a further data structure:

obtaining a value for a second level L2 in a Huffman tree, being a higher level than the first level L1;

obtaining a value identifying a first sub-tree;

identifying the node in the second level L2 of the first sub-tree, when full, corresponding to the first L2 bits of the received bit stream;

obtaining from a further data structure a data entry for the identified node, that identifies a further data structure or identifies a symbol.

27. (Cancelled)

28. (Previously Presented) A storage medium or transmission medium embodying a computer program for performing the method of claim 26.

29. (Previously Presented) Data representing a Huffman coding tree comprising leaf nodes and interior nodes arranged in H levels, wherein a leaf node depends from an interior node of the next lowest level and represents a symbol and wherein an interior node depends from an interior node of the next lowest level, the data comprising:

a first data structure identifying, for each of the nodes within a first specified level of

the tree, a symbol for each leaf node and a further data structure for each interior node, including a second data structure for a first interior node;

at least a second data structure, identified by the first data structure, identifying for each of the nodes within a sub-tree, depending from the first interior node, and at a second specified level of the tree, a symbol for each leaf node and a further data structure for an interior node, if any; and

data specifying at least the first level, the second level and the first interior node.

30. (Original) Data as claimed in claim 29, wherein the first data structure identifies a symbol for each empty node, if any.

31. (Previously Presented) Data as claimed in claim 29, wherein the second data structure identifies a symbol for each empty node of the sub-tree at a second level of the tree.

32. (Previously Presented) Data as claimed in claim 29, wherein the first level is the lowest level within the tree with at least two leaf nodes.

33. (Previously Presented) Data as claimed in claim 29, wherein the second level is the lowest level within the sub-tree with at least two leaf nodes.

34. (Previously Presented) Data as claimed in claim 29, wherein the first interior node, when at level L (L=0, 1,2..) and having a value V, is specifying by a value dependent upon V*2^(H-L).

35. (Previously Presented) Data as claimed in claim 29, further comprising data specifying H.

36. (Previously Presented) A storage medium or transmission medium embodying the data as claimed in claim 29.

37. (Original) A method of representing a Huffman binary tree comprising:

producing a first data structure associated with an identified first level L1 of the tree and comprising a plurality of data entries, each entry corresponding to a node of a full tree at the identified first level and identifying a further data structure if that node is an interior node and otherwise identifying a symbol; and

producing at least a further data structure associated with an identified second level L2 of the tree and with an identified first sub-tree and comprising a plurality of data entries, each entry corresponding to a node of the first sub tree, when full, at the second identified level L2 and identifying a further data structure if that node is an interior node and otherwise identifying a symbol.

38. (Original) A method as claimed in claim 37, running an algorithm to determine the number of data structures and their associated levels within the Huffman tree.

39. (Previously Presented) A method as claimed in claim 37 further comprising identifying a sub-tree having a root node at level L (L=0, 1,2..) and value V using a value dependent upon V*2^(H-L).

40. (Cancelled)

41. (Cancelled)

42. (Currently Amended) A decoder, comprising:
means for storing a plurality of data structures representing a Huffman coding tree of height H including at least a first data structure having an associated first offset value and an associated first shift value and a second data structure having an associated second offset value and an associated second shift value;
means for subtracting an offset value from a codeword of H bits taken from a bitstream to produce a result;
means for shifting the result by a shift value; and
means for addressing a data structure using the shifted result.

43. (Previously Presented) The decoder of claim 42, wherein the first data structure represents a first level of the Huffman coding tree and the second data structure represents a second, lower level of the Huffman coding tree.

44. (Previously Presented) The decoder of claim 43, wherein the first shift value corresponds to the first level.

45. (Previously Presented) The decoder of claim 43, wherein the second shift value corresponds to the second level.

46. (Previously Presented) The decoder of claim 43, wherein the second offset value identifies a position of a first sub-tree within the Huffman tree.

47. (Currently Amended) The decoder of claim 42, further comprising means, responsive to having obtained a value from addressing the ~~associated~~ data structure, for comparing using that value and in dependence upon a result of comparing to either use the value obtained from addressing the data structure to identify a symbol or a new current offset value.

48. (Previously Presented) The decoder of claim 47, wherein the comparing means use a most significant bit (MSB) of the value.

49. (Previously Presented) The decoder of claim 47, wherein the current offset value is initially set to the first offset value.